THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Speed-up of Stochastic Simulation of PCTMC Models by Statistical Model Reduction

Citation for published version:
Feng, C & Hillston, J 2015, Speed-up of Stochastic Simulation of PCTMC Models by Statistical Model Reduction. in *Computer Performance Engineering: 12th European Workshop, EPEW 2015, Madrid, Spain, August 31 - September 1, 2015, Proceedings.* Lecture Notes in Computer Science, vol. 9272, Springer Berlin Heidelberg, pp. 291-305, European Performance Engineering Workshop, Madrid, Spain, 31/08/15.

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Computer Performance Engineering

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 07. May. 2023

Speed-up of Stochastic Simulation of PCTMC Models by Statistical Model Reduction

Cheng Feng and Jane Hillston

LFCS, School of Informatics, University of Edinburgh, Scotland, UK
s1109873@sms.ed.ac.uk, jane.hillston@ed.ac.uk
http://www.quanticol.eu

Abstract. We present a novel statistical model reduction method which can significantly boost the speed of stochastic simulation of a population continuous-time Markov chain (PCTMC) model. This is achieved by identifying and removing agent types and transitions from the simulation which have only minor impact on the evolution of population dynamics of target agent types specified by the modeller. The error induced on the target agent types can be measured by a normalized coupling coefficient, which is calculated by an error propagation method over a directed relation graph for the PCTMC, using a limited number of simulation runs of the full model. Those agent types and transitions with minor impact are safely removed without incurring a significant error on the simulation result. To demonstrate the approach, we show the usefulness of our statistical reduction method by applying it to 50 randomly generated PCTMC models corresponding to different city bike-sharing scenarios.

1 Introduction

Continuous time Markov chains (CTMC) have been widely used to study population dynamics in many domains such as ecology [1], system biology [2] and computer networking [3]. Recently, with the widespread adoption of wireless communication techniques, large-scale collective ICT systems comprised of many communicating entities and without centralised control, have become feasible and their pervasive, transparent nature makes it crucial that their dynamic behaviour is predicted prior to deployment. Population CTMCs (PCTMC) have been proposed as a suitable tool to model such systems.

Currently, there are two typical approaches to analyse PCTMC models. One is through stochastic simulation, which is usually computationally expensive as performance metrics can only be derived after many simulation runs. The other is to build an analytical model in the form of initial value problems by fluid-limit theory [4] or moment closure techniques [5]. However, due to the intrinsic spatial-heterogeneity in collective systems (the same agent can exhibit different behaviour in different positions), the analytical model can be unresolvable due to the number of coupled ODEs in the model. Moreover, the question also arises of whether the fractured population is large enough to justify fluid/moment closure techniques [6]. As a result, in many circumstances, stochastic simulation is the

only option to analyse such models. Nevertheless, although the spatial property of collective systems increases the complexity of the underlying PCTMC models, on the other hand, we also find that it gives us the possibility to decouple parts of the model. For instance, two agents which are located far away from each other are generally less likely to influence each other than another two agents located in close proximity. In this paper, we propose a statistical model reduction method which can significantly boost the speed of stochastic simulation of PCTMC models by identifying and removing those unimportant agents and transitions with respect to some target agents which we are interested in, after a few simulation runs. The error caused by removing these agents and transitions can be controlled by the modeller by setting an acceptable error threshold.

Specifically, in order to evaluate the coupling coefficient between two agent types (which tell us how much error will be caused to the population dynamics of one agent type if we discard agents of the other type and their associated transitions), we build a directed relation graph (DRG) for the PCTMC model. We are inspired by the DRG first introduced by Lu and Law [7] for species and reaction reduction in the numerical simulation of chemical reaction mechanisms for hydrocarbon oxidation. This approach has since been improved by many researchers in the combustion research domain. Examples include DRG with error propagation [8], DRG with sensitivity analysis [9], etc.

In our DRG for PCTMC models, the vertices are the agent types and the directed edges are coupling coefficients between agent types. In the DRG for species in chemical reaction mechanisms for hydrocarbon oxidation, the technique is used to reduce a deterministic model (a set of coupled ODEs), and the coupling coefficients between species can be computed using experimental data. In contrast, in our work we reduce a stochastic model when there is no experimental data available to establish the value of the coupling coefficients. Instead, we designed two statistical reduction algorithms in which we evaluate the coupling coefficients between agent types based on a limited number of simulation runs of the full PCTMC model. After that, an error propagation method is applied to identify those agent types and transitions which can be discarded without leading to significant error. These agents are removed from the simulation, and the remaining model contains only necessary agent types and transitions, which are tightly coupled to the identified target agents. The whole process is fast, and has low computational cost compared to the total simulation cost.

To demonstrate the efficiency of this method, we will apply it to a family of 50 models for bike-sharing systems based on random topology of stations and random values of parameters. The model is specified in PALOMA [10], a process algebra recently designed for the modelling of collective adaptive systems, which makes it easy to generate many variations of the same model. The underlying PCTMC model can be automatically generated according to the population semantics of this modelling language. Note that although we illustrate the application of our method on PCTMC models derived from PALOMA, our method is not limited to PALOMA models. Moreover, to our knowledge, this is the first work which applies the DRG method outside the combustion simulation domain.

2 A Brief Introduction of PCTMC

A CTMC is a Markovian stochastic process defined on a finite state space and evolving in continuous time. In this paper, we specifically consider PCTMC models of interacting agents [11], in which we assume that there are a number of distinct agent types, each of which has a potential population. Agents interact via a set of transitions. Transitions will change one or more agents from one type to another. In general, a PCTMC model can be expressed as a tuple $\mathcal{P} = (\mathbf{X}, E, \mathbf{x_0})$:

- $\mathbf{X} = (x_1, ..., x_n) \in \mathbb{Z}^n_{\geq 0}$ is an integer vector with the ith $(1 \leq i \leq n)$ component representing the current population level of an agent type i.
- $E = \{e_1, ..., e_m\}$ is the set of transitions, of the form $e = (r_e(\mathbf{X}), \mathbf{d}_e)$, where:
  1. $r_e(\mathbf{X}) \in \mathbb{R} \geq 0$ is the rate function, associating with each transition the rate of an exponential distribution, depending on the global state of the model.
  2. $\mathbf{d}_e \in \mathbb{Z}^n$ is the update vector which gives the net change for each element of $\mathbf{X}$ caused by transition e.
- $\mathbf{x_0} \in \mathbb{Z}^n_{\geq 0}$ is the initial state of the model.

Transition rules can be easily visualised in the chemical reaction style, as

$$x_1 + \ldots + x_n \rightarrow (x_1 + d^1_e) + \ldots + (x_n + d^n_e) \quad \text{at } r_e(\mathbf{X})$$

where d^i_e $(1 \leq i \leq n)$ denotes the net change on the population of agents of type i caused by transition e. The tuple $\mathcal{P}$ contains all the information that is needed for the discrete event simulation of a PCTMC model using standard simulation algorithms, like SSA [12]. Clearly, the speed of stochastic simulation is dependent on the number of agent types, the populations of agents and the number of transitions in the model.

3 Directed Relation Graph with Error Propagation

The DRG method with error propagation was proposed in [8] to efficiently recognise removable species and reactions in the numerical simulation of large scale chemical kinetic mechanisms. In this section, we will focus on the modification of this method for application to PCTMC models.

Specifically, in the DRG for a PCTMC model, each vertex represents an agent type in the PCTMC. There exists an edge from vertex i to vertex j if and only if the removal of agents of type j and their associated transitions would directly induce an error in the evolution of the population dynamics of agents of type i. This effect can be quantified by a normalized coupling coefficients c_{ij}, defined as

$$c_{ij} = \frac{|\sum_{e \in E} r_e \;\; d^i_e \;\; \delta^j_e|}{\max(Prod_i, Cons_i)} \quad (1)$$

where r_e is the rate of transition e, d^i_e is the net change to the population of agent type i caused by transition e, δ^j_e equals 1 if agent type j is involved in

transition e otherwise it is 0 (we say agent type j is involved in transition e if and only if the net change on the population of agents of type j caused by transition e is non-zero, or the rate of transition e depends on the population of agents of type j), and

$$Prod_i = \sum_{e \in E} \mathbf{1}_{d_e^i > 0}\, r_e\, d_e^i \quad (2)$$

$$Cons_i = -\sum_{e \in E} \mathbf{1}_{d_e^i < 0}\, r_e\, d_e^i \quad (3)$$

which are the total production and consumption of agents of type i respectively. c_{ij} is bounded between 0 and 1 since

$$|\sum_{e \in E} r_e\ d_e^i\ \delta_e^j| = |\sum_{e \in E} \mathbf{1}_{d_e^i > 0}\, r_e\, d_e^i\, \delta_e^j + \sum_{e \in E} \mathbf{1}_{d_e^i < 0}\, r_e\, d_e^i\, \delta_e^j| = |Prod_{ij} - Cons_{ij}| \quad (4)$$

where $Prod_{ij}$ ($Cons_{ij}$) is the total production (consumption) of agents of type i from transitions in which agent type j is involved. Then, as $0 \leq Prod_{ij} \leq Prod_i$ and $0 \leq Cons_{ij} \leq Cons_i$, it can be inferred that $-Cons_i \leq Prod_{ij} - Cons_{ij} \leq Prod_i$, which is equivalent to $|Prod_{ij} - Cons_{ij}| \leq \max(Prod_i, Cons_i)$.

Furthermore, we say agent type j and its associated transitions are *removable* if $c_{ij} \leq \theta$, where i is the target agent type which we are interested in and θ is an *acceptable error threshold* given by the modeller. Moreover, note that coupling coefficients are not symmetric, since it is not necessarily the case that $c_{ij} = c_{ji}$.

3.1 Group-based Direct Coupling Coefficient

We will remove agents (and their transitions) one by one until the cumulative induced error reaches the acceptable error threshold. Since a transition typically involves more than one agent type, when we consider removing an agent type, we cannot assume that it is independent of the agent types that have already been removed. The following equation gives the coupling coefficient of an agent type which also takes into account those agent types which have already been removed:

$$c_{ij,\{S\}} = \frac{|\sum_{e \in E} r_e\ d_e^i\ \delta_e^{j,\{S\}}|}{\max(Prod_i, Cons_i)} \quad (5)$$

where S is the set of agent types which have already been removed, $\delta_e^{j,\{S\}}$ equals 1 as long as agent type j or an agent type in S is involved in transition e, otherwise it is 0.

3.2 Indirect Coupling Coefficient

For those agent types which are not directly connected in the DRG, by using an error propagation method, we can evaluate the indirect coupling coefficient

between two agent types. Specifically, indirect coupling is quantified by path dependent coefficient $c_{ij,\sigma}$, which is the product of the direct coupling coefficients along the path σ between agent types i and j. The influence of removing agent type j on agent type i is characterized by coefficient C_{ij}, which is the maximum of the path dependent coefficients:

$$c_{ij,\sigma} = \prod_{xy \in \sigma} c_{xy} \quad (6)$$

$$C_{ij} = \max_{all\ paths\ \sigma} c_{ij,\sigma} \quad (7)$$

Figure 1 shows part of a DRG, in which the indirect coupling coefficient between i and j is $C_{ij} = c_{im} \times c_{mj} = 0.24$, since $c_{im} \times c_{mj} > c_{ik} \times c_{kl} \times c_{lj}$.

Similarly, taking into consideration agent types that have already been removed, we define:

$$c_{ij,\{S\},\sigma} = \prod_{xy \in \sigma} c_{xy,\{S\}} \quad (8)$$

$$C_{ij,\{S\}} = \max_{all\ paths\ \sigma} c_{ij,\{S\},\sigma} \quad (9)$$

where $C_{ij,\{S\}}$ is the indirect coupling coefficient from agent type i to j given a set of agent types $\{S\}$ which have already been removed.

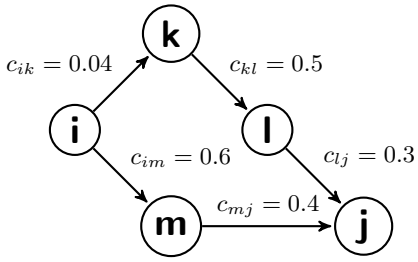

Fig. 1. Part of a directed relation graph with five agent types

4 Statistical Model Reduction

In this section, we give the details of our statistical model reduction method. Given the target agent types which we are going to investigate, the dynamics in a PCTMC and the acceptable error threshold for those agent types, after a few simulation runs of the full model our method can accurately identify those agent types and transitions which can be removed from the simulation without leading to an error beyond the acceptable threshold, based on the DRG with error propagation. For simplicity, we only deal with a single target agent type in the introduction of the two algorithms, but it can be readily seen that they can both be extended to cope with multiple target agent types.

4.1 Statistical Transition Rate Evaluation

First of all, since the rate of a transition, $r_e(\mathbf{X})$ depends on the current state of the PCTMC, we cannot evaluate c_{ij} in Equation (1) solely from the system description since r_e may evolve during the simulation period. However, by learning from past simulation runs, we can give a statistical rate for each transition. Specifically, we let $r_e^k = \frac{N_e^k}{T}$, where N_e^k is the firing count of transition e during the kth simulation run, T is the time length of a simulation run, r_e^k is the statistical rate of transition class e in the kth simulation. We use the statistical transition rate to estimate the coupling coefficient, which gives the overall contribution of an agent type to the evolution of the population dynamics of the target agent type during a simulation.

4.2 Model Reduction Algorithms

We introduce two automatic model reduction algorithms both based on DRG with error propagation, but with different sampling methods for deducing the coupling coefficients and different reduction criteria.

Algorithm with Fixed Length Sampling We first introduce a straightforward reduction algorithm. Specifically, we let the user define a number K, which is the number of simulation runs before the reduction process is actually carried out. At the end of the Kth simulation run, we approximate the rate of transition e by $r_e = \frac{\sum_{k=1,\ldots,K} N_e^k}{K \times T}$, i.e. the average of the statistical transition rate over the K runs. Then, we initiate the DRG and compute the coupling coefficients for the target agent type with respect to all other agent types. We delete an agent type which has the minimal coupling coefficient with the target agent type each time, and then update the coupling coefficients of remaining agent types, taking into account the newly removed agent type. The reduction process is terminated when removing an agent type will exceed the acceptable error threshold given by the user. Algorithm 1 gives the pseudo code for this algorithm.

Algorithm with Flexible Length Sampling The second algorithm is more strict in the sense that we only remove agent types after we reach $\Pr(C_{ti,\{S\}} < \theta) \geq p$, where p is the confidence level. That is to say, that given a set of already removed agents $\{S\}$, we only remove agent type i when we achieve confidence level p that the coupling coefficient $C_{ti,\{S\}}$ is less than the acceptable error threshold θ.

In order to test whether $\Pr(C_{ti,\{S\}} < \theta) \geq p$ holds, we raise two hypotheses, which are: H_0: $\Pr(C_{ti,\{S\}} < \theta) \geq p_0$ and H_1: $\Pr(C_{ti,\{S\}} < \theta) \leq p_1$ where (p_1, p_0) is an indifference region where we say that the probability is sufficiently close to p that we are indifferent with respect to which of the two hypotheses H_0 or H_1 is accepted. Moreover, we require the probability of accepting H_1 when actually H_0 holds to be less than α, and the probability of accepting H_0 when actually H_1 holds to be less than β.

Algorithm 1 Reduction with fixed length sampling

User specifies the target agent type t, number of sampling simulation runs K and the acceptable error threshold θ.
Simulate the full model K runs
Compute statistical rates for all transitions.
Initiate the DRG, and compute the coupling coefficients from the target agent type to any other agent types.
repeat
  find agent type i which has the minimal coupling coefficient $C_{ti,\{S\}}$ among the remaining agent types, where $\{S\}$ is set of removed agent types.
  if $C_{ti,\{S\}} < \theta$ **then**
    remove agent type i and the transitions it is involved.
    add agent type i to $\{S\}$, and update the coupling coefficient of remaining agent types.
  end if
until $C_{ti,\{S\}} \geq \theta$
Use the reduced model in the following simulation runs.

Furthermore, we let

$$\frac{p_{1m}}{p_{0m}} = \frac{p_1^{d_m}(1-p_1)^{m-d_m}}{p_0^{d_m}(1-p_0)^{m-d_m}}$$

where m is the current number of simulation runs, d_m is the number of simulation runs for which $C_{ti,\{S\}} < \theta$ holds. Applying the idea behind the sequential probability ratio test (SPRT) [13], we can accept H_0 when $\frac{p_{1m}}{p_{0m}} \leq \frac{\beta}{1-\alpha}$ and accept H_1 when $\frac{p_{1m}}{p_{0m}} \geq \frac{1-\beta}{\alpha}$. Accepting H_0 means that with confidence level p removing agent type i and the associated transitions will cause error in the target agent type of less than θ, thus the agent type and transitions are removable. Accepting H_1 means that we are highly confident that we cannot remove agent type i, thus the reduction process should be terminated.

It can be seen that we can only reach the removal criterion of an agent type, or the termination criterion of the reduction process, after at least $K = \min(\log_{\frac{p_1}{p_0}} \frac{\beta}{1-\alpha}, \log_{\frac{1-p_1}{1-p_0}} \frac{1-\beta}{\alpha})$ simulation runs, when $d_m = m$ or $d_m = 0$. Thus, we will start the reduction process after K simulation runs. Specifically, at the end of the Kth simulation run, we compute the average statistical rate for each transition as $r_e = \frac{\sum_{k=1,\ldots,K} N_e^k}{K \times T}$, and then build the DRG and compute the coupling coefficients using the average statistical transition rates. Again, we find the agent type which has the minimal coupling coefficient with the target agent type. Then, we remove an agent type each time repeatedly if hypothesis H_0 holds. We simulate the model with the reduced version. Compared with the first algorithm, the difference is that we will continue to check whether there are more agent types that are removable until we reach hypothesis H_1. Note that when some agent types and transitions are removed, we lose information about the rate of those removed transitions in the following runs. Thus, we will use

Algorithm 2 Reduction with flexible length sampling

User specifies t, θ, $p0$, $p1$, α and β
Simulate the full model $K = \min(\log_{\frac{p_1}{p_0}} \frac{\beta}{1-\alpha}, \log_{\frac{1-p_1}{1-p_0}} \frac{1-\beta}{\alpha})$ runs
Compute statistical transition rates for each simulation run, and the average statistical transition rates for the K runs.
Initiate the DRG and compute the coupling coefficients from the target agent type t to any other agent types using the average statistical transition rates.
Point A:
repeat
  find agent type i which has the minimal coupling coefficient $\overline{C}_{ti,\{S\}}$ using the average statistical transition rates among the remaining agent types, where $\{S\}$ is set of removed agent types.
  Compute $C_{ti,\{S\}}$ for each simulation run
  if $\frac{p_{1m}}{p_{0m}} \leq \frac{\beta}{1-\alpha}$ **then**
    remove agent type i and the transitions it is involved.
    add agent type i to $\{S\}$, and update the coupling coefficient of remaining agent types using the average statistical transition rates.
  else if $\frac{p_{1m}}{p_{0m}} \geq \frac{1-\beta}{\alpha}$ **then**
    stop the reduction process.
  end if
until $\frac{p_{1m}}{p_{0m}} > \frac{\beta}{1-\alpha}$
Simulate the reduced model, after each run go to Point A unless the reduction process has stopped.

the average rate of those removed transitions in the previous simulation runs in order to calculate the coupling coefficients if they are needed. Algorithm 2 gives the pseudo code for this algorithm.

4.3 Comparison of the Two Algorithms

It is obvious that the algorithm with fixed length sampling is easier to implement and has less computational cost for the reduction process. However, the algorithm with flexible length sampling has the advantage of more stringent error control on the simulation result. We will report on a comprehensive test of these two reduction algorithms in Section 6, using 50 randomly generated city bike-sharing models. But before that, we first briefly introduce the modelling language we use to generate our models.

5 Modelling Language and Model Definition

In this section, we give a brief introduction of the modelling language, PALOMA, and the model definition of the bike-sharing scenario using this language.

5.1 PALOMA

PALOMA is a stochastic process algebra, specifically designed to support the construction of formal models of large collective adaptive systems in which agents

are distributed over a discrete set of named locations, $\mathcal{L}$. Agents are parameterised by a *location*, denoted by ℓ, $\ell \in \mathcal{L}$. There is a finite set of action types $\mathcal{A}$, and actions may be undertaken spontaneously or may be induced by a message of the same type, sent by another agent in the system. All spontaneous actions are assumed to have a duration governed by an exponential distribution and characterised by a rate r. A model P consists of a number of agents composed in parallel. The language has the following grammar:

$$\begin{aligned} \pi &::= !(\alpha, r)@\mathbf{IR}\{\overrightarrow{\ell}\} \mid ?(\alpha, p)@\mathbf{Pr}\{v\} \mid !!(\alpha, r)@\mathbf{IR}\{\overrightarrow{\ell}\} \mid ??(\alpha, p)@\mathbf{Wt}\{v\} \mid (\alpha, r) \\ S(\ell) &::= \pi.S'(\ell') \mid S_1(\ell) + S_2(\ell) \mid M \\ P &::= S(\ell) \mid P \parallel P \end{aligned}$$

Agents can change their states and locations by different actions:

Spontaneous action with broadcast message emission: $!(\alpha, r)@\mathbf{IR}\{\overrightarrow{\ell}\}$ describes that the agent performs an action α, $\alpha \in \mathcal{A}$, *spontaneously* with rate r. During the occurrence of the action, a broadcast message, also typed α, is emitted. The *influence range* of the broadcast is defined by the location vector $\overrightarrow{\ell}$, which gives a list of locations where agents can *potentially* be influenced by this message. For example, $\overrightarrow{\ell} = range(d)$ denotes that the influence range is a set of locations whose distance from the location of the sender agent is less than a specific threshold d. Another frequently used definitions of influence range is $\overrightarrow{\ell} = local$, which represent that the influence range of the broadcast message is restricted to the location of the sender agent.

Spontaneous action with unicast message emission: $!!(\alpha, r)@\mathbf{IR}\{\overrightarrow{\ell}\}$ also describes a spontaneous action of type α, rate r and influence range $\overrightarrow{\ell}$. The difference is that here the message is a unicast, meaning that at most one agent can receive the message.

Action induced by a broadcast message: $?(\alpha, p)@\mathbf{Pr}\{v\}$ describes that the agent performs an action α *immediately* after receiving and accepting a broadcast message of type α. Whether the agent receives the broadcast message is decided by two factors. Firstly, the agent must be located within the influence range of the message; otherwise, the message will be ignored. Secondly, the value $v \in [0, 1]$ gives the probability that the message is received by the agent given that it is within the influence range of the broadcast. v can be defined dynamically. For instance, $v = 1/|S(\ell)|$ denotes that the message reception probability is dependent on the number of agents in state S in location ℓ, where $|\cdot|$ is an operator which gives the number of agents in a particular state and location. Formally, the definition of v follows this grammar:

$$v ::= c \quad | \quad dist(\ell_1, \ell_2) \quad | \quad |S(\ell)| \quad | \quad v\ (op)\ v$$

where c is a constant real number, $dist(\ell_1, \ell_2)$ is the distance between locations ℓ_1 and ℓ_2, (op) is a basic arithmetic operator. Once the message has been received, the agent decides whether to accept it. Here, a constant value $p \in [0, 1]$ encodes the probability that the agent will accept the message. This can be thought of as the agent choosing to respond to a spontaneous action of the given type with

probability p. The definition of v and p supports a rich set of possible interaction patterns between agents.

Action induced by a unicast message: $??(\alpha, p)@\mathbf{Wt}\{v\}$ describes that the agent performs an action α immediately after receiving and accepting a unicast message of type α. Here, $v \in \mathbb{R}^+$ gives the *weight* of the agent to be the receiver of this unicast message. The definition of v follows the same grammar as previously, but with a different value domain. The weights are used to resolve between several potential receiver agents: suppose there are n agents denoted by $S_1(\ell_1), S_2(\ell_2), ..., S_n(\ell_n)$, which can potentially receive the unicast message, with weights $v_1, v_2, ..., v_n$. Then, the probability that agent $S_1(\ell_1)$ receives the message is v_1/Σ, where Σ denotes $\sum_{i=1}^{n} v_i$, the sum of the associated weights of all potential receivers. If there is no potential receiver, the message is simply discarded. The value $p \in [0, 1]$ is a distinct probability deciding whether a received message is accepted or not. Note that if the selected agent does not accept the unicast message, the message is discarded; it cannot be passed to any other potential receiver agent.

Spontaneous action without message emission: (α, r) denotes that the agent performs a spontaneous action named α with a rate r governed by a negative exponential distribution. No message is sent out during the firing of this action.

Alternative behaviours are represented by the standard choice operator, $+$. A choice between spontaneous actions is resolved via the race policy, based on their corresponding rates. A choice between two induced actions of the same type within a single component is not allowed. M denotes a constant name for an agent. Compositionality is proved by the parallel operator.

5.2 Model Definition

Here we present the PALOMA model for the template city bike-sharing scenario which can be automatically parsed to a PCTMC model via the population semantics introduced in [10]. Suppose that there are n bike stations in the city, and each one has a number of available bikes and slots. Therefore, we represent the available bikes and slots in Station i (for $i = 1, ..., n$) by agents as follows:

$$Slot(\ell_i) = ??(return, 1)@\mathbf{Wt}\{1\}.Bike(\ell_i) \quad Bike(\ell_i) = ??(borrow, 1)@\mathbf{Wt}\{1\}.Slot(\ell_i)$$

Both $Slot(\ell_i)$ and $Bike(\ell_i)$ are passive. They can only be induced to make a *return* (returning a bike to this station) or *borrow* (borrowing a bike from this station) action by a unicast message, and when this happens they switch role.

The agents to represent the bike stations are defined as:

$$\begin{aligned} Station(\ell_i) = {} & !(SlotAvailable_i, \gamma)@\mathbf{IR}\{range(1)\}.Station(\ell_i) + \\ & !(BikeAvailable_i, \gamma)@\mathbf{IR}\{range(1)\}.Station(\ell_i) \end{aligned}$$

A bike station performs both $BikeAvailable_i$ and $SlotAvailable_i$ self-jump spontaneous actions with broadcast message emission at the rate of γ. The influence range of the broadcast messages is defined by the function $range(d)$, which means that only agents in locations whose distance to the location of the sender station is less than d can potentially be influenced by this message.

The agents representing bike users are defined as follows:

$$
\begin{aligned}
\mathit{Pedestrian}(\ell_i) &= (\mathit{seekb}_i, b_i).\mathit{SeekBike}(\ell_i) + \sum_{j\neq i}(\mathit{walk}_{ij}, w_{ij}).\mathit{Pedestrian}(\ell_j)\\
\mathit{SeekBike}(\ell_i) &= \sum_{j=1}^{m} ?(\mathit{BikeAvailable}_j, 1)@\mathbf{Pr}\{v_1\}.\mathit{Walk2Station}_j(\ell_i)\\
\mathit{Walk2Station}_j(\ell_i) &= (\mathit{W2S}_{ij}, w2s_{ij}).\mathit{CheckBikeNum}(\ell_j)\\
\mathit{CheckBikeNum}(\ell_i) &= ?(\mathit{BikeAvailable}_i, 1)@\mathbf{Pr}\{v_2\}.\mathit{BorrowBike}(\ell_i)\\
\mathit{BorrowBike}(\ell_i) &= !!(\mathit{borrow}, o)@\mathbf{IR}\{\mathit{local}\}.\mathit{Biker}(\ell_i)\\
\mathit{Biker}(\ell_i) &= (\mathit{seeks}_i, s_i).\mathit{SeekSlot}(\ell_i) + \sum_{j\neq i}(\mathit{ride}_{ij}, r_{ij}).\mathit{Biker}(\ell_j)\\
&\cdots\\
\mathit{ReturnBike}(\ell_i) &= !!(\mathit{return}, o)@\mathbf{IR}\{\mathit{local}\}.\mathit{Pedestrian}(\ell_i)
\end{aligned}
$$

where

$$v_1 = \theta_0 + \theta_1 \frac{d - \mathit{dist}(\ell_i, \ell_j)}{d} + \theta_2 \frac{|\mathit{Bike}(\ell_j)|}{|\mathit{Bike}(\ell_j)| + |\mathit{Slot}(\ell_j)|} \tag{1}$$

$$v_2 = \frac{|\mathit{Bike}(\ell_i)|}{|\mathit{Bike}(\ell_i)| + \sigma} \tag{2}$$

As can be seen from the definition, when the user agent is in the *Pedestrian* state, it travels from location ℓ_i to location ℓ_j at the rate of w_{ij} by performing a spontaneous action walk_{ij} without message emission. It may also seek a bike at the rate of b_i, and enter into the *SeekBike* state.

The user agent in the $\mathit{SeekBike}(\ell_i)$ state can do a $\mathit{BikeAvailable}_j$ action induced by a broadcast message sent by a station agent in location ℓ_j and goes to the $\mathit{Walk2Station}_j(\ell_i)$ state, which represents that the user is walking from location ℓ_i to the bike station in location ℓ_j. The probability of receiving a bike available message from the station in location ℓ_j is defined in Equation (1). It can be interpreted as follows: the users tend to borrow a bike from a closer bike station with more available bikes, and θ_1, θ_2 are associated weights of those factors, θ_0 is the noise term (imagine that the user checks the bike numbers in nearby stations using a smart phone application). The user in the $\mathit{Walk2Station}_j(\ell_i)$ state can do a spontaneous action $\mathit{W2S}_{ij}$ at the rate of $w2s_{ij}$, where $1/w2s_{ij}$ is the expected time to walk from ℓ_j to the bike station in ℓ_i.

The user in the $\mathit{CheckBikeNum}(\ell_i)$ state can only do a $\mathit{BikeAvailable}_i$ action induced by a broadcast message sent by the station in ℓ_i. The probability of receiving the message is defined in Equation (2), where σ is a very small real number to avoid a zero denominator. This ensures that the user can only go to the $\mathit{BorrowBike}(\ell_i)$ state if the bike station is not empty. The borrow bike action *borrow* is fired at rate o. Meanwhile, a unicast message *borrow* is sent out, and the user becomes a *Biker*.

A user agent in the *Biker* state can perform actions and become a *Pedestrian* again in a similar fashion, thus we do not give the details due to lack of space.

Finally, the initial population of agents are given in the following definition:

$$\ldots \parallel \mathit{Pedestrian}(\ell_i)[n_p^i] \parallel \mathit{Slot}(\ell_i)[n_s^i] \parallel \mathit{Bike}(\ell_i)[n_b^i] \parallel \mathit{Station}(\ell_i) \parallel \ldots$$

where $\mathit{Pedestrian}(\ell_i)[n_p^i]$ is syntactic sugar which represents n_p^i copies of $\mathit{Pedestrian}(\ell_i)$ in parallel.

6 Experiments

The usefulness of a reduction algorithm can be evaluated by the size of the reduced model (the proportion of removed agent types and transitions), the decrease of simulation time, and the error caused by the reduction. Thus, in order to do the evaluation, we simulate the bike-sharing models with and without applying the reduction algorithms. To make our experiments more thorough, we generated 50 bike-sharing models each with 30 locations. There are 50 pedestrians and a bike station which is equipped with 25 available bikes and 5 available slots initially in each location in the simulation. The topology of the locations and the value of parameters in each model are generated randomly.

We simulate each model without reduction for 500 runs. Next, we randomly pick the bike agents in 2 bike stations as our target agent types, denoted as t_1 and t_2. The two reduction algorithms are applied in the simulation of these models with different acceptable error thresholds with respect to t_1 and t_2. For each value of the acceptable error threshold, we also simulate each model for 500 runs (including the sampling runs), and compare the size of the reduced model and the decrease of simulation time with the full model without reduction. In the simulation with the reduction algorithm with fixed length sampling, we set the sampling length to 50 simulation runs. In the simulation with the reduction algorithm with flexible length sampling, we set $p0 = 0.95$, $p1 = 0.9$, $\alpha = \beta = 0.1$.

Figure 2 gives the proportional reduction of simulation time, agent types and transitions with different reduction algorithms and varying acceptable error thresholds. It can be seen that both reduction algorithms can significantly reduce the size of the model and simulation time. Observe that the larger the value of the error threshold, the more agent types, transitions and simulation time can be reduced within the model. This reflects the soundness of our reduction algorithms from another perspective. Moreover, it can be seen that the algorithm with fixed length sampling tends to remove more agent types and transitions from the simulation. The reduction in simulation time cost when applying the flexible sampling method is smaller compared with fixed length sampling method both due to a proportionally smaller reduction of agent types and transitions, and the larger computational cost of the reduction process.

Furthermore, in order to measure the error caused by reduction, we evenly sample the population of target agents at 200 time points along the simulation. The error in the population of a target agent type t at a time point i can be quantified by: $\mathit{Error}_{t,i} = \frac{|\,\overline{x^f_{t,i}} - \overline{x^r_{t,i}}\,|}{\overline{x^f_{t,i}}}$ where $\overline{x^r_{t,i}}$ and $\overline{x^f_{t,i}}$ are the average population of agents of type t at time point i in the 500 simulation runs with and

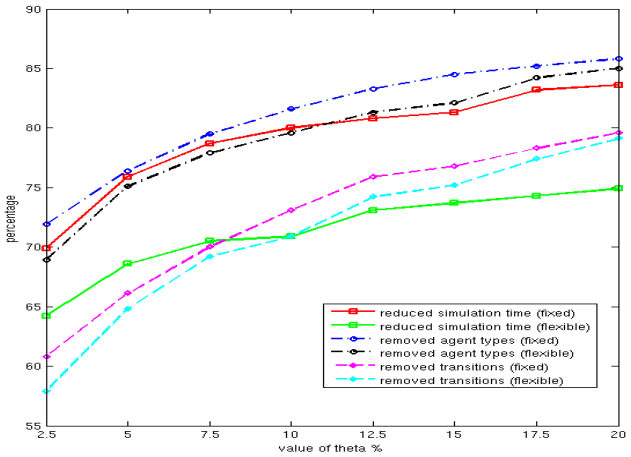

Fig. 2. The proportional reduction of simulation time, agent types and transitions (y-axis) with different reduction algorithms and acceptable error thresholds (x-axis).

Value of θ	2.5%	5%	7.5%	10%	12.5%	15%	17.5%	20%
Fixed Length Sampling	1.5%	2.8%	3.1%	3.4%	3.5%	3.9%	4.1%	4.6%
Flexible Length Sampling	1.1%	1.8%	2.9%	3.0%	3.3%	3.8%	4.1%	4.4%

Table 1. The average error caused by reduction with different algorithms and acceptable error thresholds.

without reduction. If we treat each $Error_{t,i}$ where $t \in \{t_1, t_2\}$, $i \in (1, 2, \ldots, 200)$ as an error sample, then the average error caused by a reduction algorithm in our experiments can be measured by:

$$\overline{Error_{t1,t2}} = \frac{\sum_{i=1}^{200}(Error_{t1,i} + Error_{t2,i})}{2 \times 200}$$

where $t1$, $t2$ are the target agent types in our experiments. Table 1 gives the average error caused by reduction with different algorithms and acceptable error thresholds. Table 2 shows the 95th percentiles of the error samples (95% of the error samples are below this value). We find that the average error caused by both reduction algorithms is significantly smaller than the acceptable error threshold we assign to the target agents. We can also observe that the reduction algorithm with flexible length sampling has better performance in controlling the error in the tail than the algorithm with fixed length sampling.

6.1 Discussion

As modellers we know that a model is inevitably an abstraction of the system in the real world. Thus it inevitably contains some deviation from the real system,

Value of θ	2.5%	5%	7.5%	10%	12.5%	15%	17.5%	20%
Fixed Length Sampling	2.8%	4.7%	6.8%	8.4%	11.9%	15.8%	16.2%	18.2%
Flexible Length Sampling	1.6%	4.3%	6.3%	7.1%	9.9%	11.6%	13.6%	16.7%

Table 2. The 95th percentiles of error caused by reduction with different algorithms and acceptable error thresholds.

due to details that are omitted in the abstraction process. Consequently, except for the case of particular safety critical systems, it is generally acceptable to allow some minor noise to be introduced to a model during construction. Taking this perspective a little further, we can consider the agent types and transitions that we removed from the simulation using our reduction algorithms as noise factors which have negligible impact on the evolution of target agent types. Thus our method can significantly improve the efficiency of analysing the model whilst retaining a reasonable level of faithfulness with respect to the modelled system.

We anticipate that the benefit to be gained from our approach could be particularly valuable in statistical model checking since it usually requires thousands of simulation runs in order to check whether a hypothesis holds. For example, for the bike-sharing system, suppose we want to check whether the following hypothesis holds: $\Pr(\mathbf{G}_{[0,100]}0 < x_b < C) \geq 95\%$ where x_b is the number of bike agents in a station, C is the capacity of that station. This means we require that in the first 100 time points, the probability of the station being empty or full should be less than 5%. Thus, if we set the bike agents in that station as our target agent type, the simulation speed can be significantly boosted by removing those agent types and transitions that are loosely-coupled to the target agent type, as illustrated by the sample simulation runs presented in the previous section.

Moreover, when applying the reduction algorithm with flexible length sampling in the bike-sharing model, we find that in general more than 90% removable agent types are identified at the end of $\log \frac{p_1}{p_0} \frac{\beta}{1-\alpha}$ simulation runs which is the minimal number of runs to reach the removal criterion. Thus, identifying removable agent types and transitions should in general be much quicker than reaching the criterion to accept or reject a hypothesis in statistical model checking. We plan to explore and exploit this promising application of our approach in future work.

7 Conclusion

We have presented two statistical model reduction algorithms for the stochastic simulation of PCTMC models. Both algorithms are based on investigating the coupling coefficients between agent types in the model by building a directed relation graph and applying an error propagation method to measure agent types which are not directly related. We have shown that our reduction algorithms can significantly reduce the computational cost of the simulation. Moreover, the error caused by the reduction is well-controlled by the acceptable error threshold set by

the modeller. We have proposed that our reduction method could be very useful in statistical model checking for PCTMC models. We are going to investigate this idea further in the near future.

Acknowledgement

The authors would like to thank Daniël Reijsbergen, Vashti Galpin and Stephen Gilmore for their helpful comments on an earlier draft of this work. This work is supported by the EU project QUANTICOL, 600708.

References

1. Allen, L.J., Allen, E.J.: A comparison of three different stochastic population models with regard to persistence time. Theoretical Population Biology **64**(4) (2003) 439–449
2. Calder, M., Vyshemirsky, V., Gilbert, D., Orton, R.: Analysis of signalling pathways using continuous time Markov chains. In: Transactions on Computational Systems Biology VI. Springer (2006) 44–67
3. Cerotti, D., Gribaudo, M., Bobbio, A.: Markovian agents models for wireless sensor networks deployed in environmental protection. Rel. Eng. & Sys. Safety **130** (2014) 149–158
4. Tribastone, M., Gilmore, S., Hillston, J.: Scalable differential analysis of process algebra models. IEEE Transactions on Software Engineering **38**(1) (2012) 205–219
5. Guenther, M.C., Stefanek, A., Bradley, J.T.: Moment closures for performance models with highly non-linear rates. In: Computer Performance Engineering. Springer (2013) 32–47
6. Hillston, J.: Challenges for quantitative analysis of collective adaptive systems. In: 8th International Symposium on Trustworthy Global Computing. (2013) 14–21
7. Lu, T., Law, C.K.: A directed relation graph method for mechanism reduction. Proceedings of the Combustion Institute **30**(1) (2005) 1333–1341
8. Pepiot-Desjardins, P., Pitsch, H.: An efficient error-propagation-based reduction method for large chemical kinetic mechanisms. Combustion and Flame **154**(1) (2008) 67–81
9. Niemeyer, K.E., Sung, C.J., Raju, M.P.: Skeletal mechanism generation for surrogate fuels using directed relation graph with error propagation and sensitivity analysis. Combustion and flame **157**(9) (2010) 1760–1770
10. Feng, C., Hillston, J.: PALOMA: A process algebra for located Markovian agents. In: Quantitative Evaluation of Systems. Springer (2014) 265–280
11. Bortolussi, L., Hillston, J., Latella, D., Massink, M.: Continuous approximation of collective system behaviour: A tutorial. Performance Evaluation **70**(5) (2013) 317–349
12. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry **81**(25) (1977) 2340–2361
13. Younes, H.L., Simmons, R.G.: Probabilistic verification of discrete event systems using acceptance sampling. In: Computer Aided Verification, Springer (2002) 223–235